Analysis of Meeting Protocols by Formalisation, Simulation and Verification

Jonker, C.M.; Schut, M.C.; Treur, J.; Yolum, P.

published in
Computational and Mathematical Organization Theory
2007

DOI (link to publisher)
10.1007/s10588-006-9001-8

document version
Publisher's PDF, also known as Version of record

Comput Math Organiz Theor (2007) 13:283–314
DOI 10.1007/s10588-006-9001-8

Analysis of meeting protocols by formalisation, simulation, and verification

Catholijn M. Jonker · Martijn C. Schut · Jan Treur · Pınar Yolum

Published online: 30 September 2006

Abstract Organizations depend on regular meetings to carry out their everyday tasks. When carried out successfully, meetings offer a common medium for participants to exchange ideas and make decisions. However, many meetings suffer from unfocused discussions or irrelevant dialogues. To study meetings in detail, we first formalize general properties of meetings and a generic meeting protocol to specify how roles in a meeting should interact to realize these properties. This generic protocol is used as a starting point to study real-life meetings. Next, an example meeting is simulated using the generic meeting protocol. The general properties are formally verified in the simulation trace. Next, these properties are also verified formally against empirical data of a real meeting in the same context. A comparison of the two traces reveals that a real meeting is more robust since when exceptions happen and the rules of the protocol are violated, these exceptions are handled effectively. Given this observation, a more refined protocol is specified that includes exception-handling strategies. Based

C. M. Jonker (✉)
Delft University of Technology, Man-Machine Interaction, Mekelweg 4, 2628 CD Delft,
The Netherlands
e-mail: catholijn@mmi.tudelft.nl

M. C. Schut · J. Treur
Vrije Universiteit Amsterdam, Department of Artificial Intelligence, De Boelelaan 1081a,
1081HV Amsterdam, The Netherlands

M. C. Schut
e-mail: schut@few.vu.nl

J. Treur
e-mail: treur@few.vu.nl

P. Yolum
Department of Computer Engineering, Bogazici University, TR-34342 Bebek, Istanbul, Turkey
e-mail: pinar.yolum@boun.edu.tr

on this refined protocol a meeting is simulated that closely resembles the real meeting. This protocol is then validated against another set of data from another real meeting. By iteratively adding exception handling rules, the protocol is enhanced to handle a variety of situations successfully.

Keywords Meeting protocols · Dynamic properties · Temporal logical formalisation · Simulation · Verification · Organisation modeling

1 Introduction

Meetings are an integral part of every day life. Meetings are important tools in most organizations to structure decision processes and to disseminate information throughout the organization. Typically, the members of a group come together on a regular basis to inform each other of new developments, to discuss problems, and to propose solutions. While many organizations depend on face-to-face meetings, it is notoriously difficult to hold a focused and effective meeting. Informal representations make it difficult to analyze meeting protocols. Intuitively, if the rules of the participants are set correctly, meetings should proceed successfully. For example, if all the participants in a meeting are allowed to speak for only a certain time frame, then we may guarantee that a meeting will end on time. Similarly, if the chair of a meeting is not allowed to skip items on the meeting agenda, we may guarantee that all items are treated in the meeting. If the rules of a meeting can be captured, then we can develop methods to predict the meetings that will take place. Even though there is an abundant literature about guidelines on how to carry a successful meeting, these guidelines are rather informal, which makes it hard to put into practice and hard to evaluate (Wolf, 2002; Creighton, 2002).

A meeting can be effective only if the rules of the meeting, the *meeting protocol* is well-defined. This paper formalizes a domain-independent meeting protocol that can be used in various meetings. The protocol aimed for is the result of a process of iteratively revising an initial (possibly trivial) protocol to match with observed (empirical) meeting behavior. This paper demonstrates some first steps of this process:

(1) We start with a meeting protocol that is based on some meeting guidelines found in the literature and our own (intuitive) knowledge of conducting meetings;
(2) We simulate a meeting where the participants follow the meeting protocol mentioned in Step 1. During the simulation, we record the actions of the participants at each time step, which forms a simulation trace.
(3) We then observe a real meeting and again record the actions of the participants at each time step, which forms an empirical trace.
(4) Based on the discrepancies between the protocol and the observed meeting, we revise the protocol. Although it may seem counterintuitive that we start with a rigid formalization, it must be understood that the ultimate formalization will be the result of a process that includes real practiced meeting behavior. Ultimately, the formalization captures many intuitive ideas that are also mentioned in meeting guidelines. Also, the formalization eventually captures actions that need to be carried out by participants as well as constraints that each participant has to satisfy.

The main aim of this work is to understand how meeting protocols are carried out, by understanding the different flows that take place in meetings. To achieve this, we study the meeting protocol for an empirical trace as well as with a simulated trace and analyze various properties. The empirical trace is based on observations of a real meeting. The simulated trace is generated in a simulation environment where agents are assumed to follow the meeting protocol strictly.

The two traces are not only compared in terms of which of the protocol rules they satisfy (and which not) but also in terms of more global overall properties that are considered desirable for a meeting. It turns out that at a number of points in time the empirical trace does not satisfy some protocol rules, but still is not ineffective. This is due to the fact that as soon as exceptions on the protocol rules occur, participants (and in particular the chair person) undertake specific actions to get the meeting on the right track soon. Therefore an effective meeting need not satisfy all protocol rules. This gives the protocol rules the states of rules-with-exceptions, and other, more specific rules are used to handle exceptions if they occur (for example, re-opening an already closed agenda item if someone still talks about it). This new perspective has led to a second, more refined meeting protocol specification, in which for a number of rules also exception rules are added, i.e., rules about what to do if the original rule is not satisfied. This refined protocol is tested for a second empirical trace of a meeting.

The rest of this paper is organized as follows. Section 2 gives a technical background on the agent-based modeling approach used and the methodological perspective. Section 3 develops the formal generic meeting protocol. Section 4 introduces a simulation trace. Section 5 addresses the first empirical trace. Section 6 analyzes both meeting traces in terms of desired properties. Section 7 presents the refined protocol and a simulation based on this protocol. Sections 8 and 9 analyze the results for the refined protocol. Section 10 discusses the relevant literature in comparison to this work.

2 Methodology

Two main components of our methodological perspective are analysis and simulation. The analysis refers to the study of a process as a whole (e.g., an organization) as well as the individual agents inside the process.

The analysis for a process as a whole includes the identification and specification of:

(a) the dynamic properties of the overall process (in informal, semi-formal or formal manners),
(b) the dynamic properties for parts of the process recursively (e.g., departments, roles or agents),
(c) the interlevel relationships between the dynamic properties of the whole and dynamic properties of parts; e.g., the dynamic properties of the parts imply the dynamic properties of the whole.

To formally specify dynamic properties characterising dynamics of meetings, an expressive language is needed. To this end the Temporal Trace Language TTL is used (cf. Jonker and Treur, 2002); this language is briefly defined as follows. Examples of properties expressed in TTL can be found in later sections.

A *state ontology* is a specification (in order-sorted logic) of a vocabulary, i.e., a signature. A state for ontology Ont is an assignment of truth-values $\{\mathsf{true}, \mathsf{false}\}$ to the set $\mathsf{At}(\mathsf{Ont})$ of ground atoms expressed in terms of Ont. The *set of all possible states* for state ontology Ont is denoted by STATES(Ont). The set of *state properties* STATPROP(Ont) for state ontology Ont is the set of all propositions over ground atoms from At(Ont). A fixed *time frame* T is assumed which is linearly ordered. A*trace* or*trajectory* γ over a state ontology Ont and time frame T is a mapping γ : T $\rightarrow$ STATES(Ont), i.e., a sequence of states γ_t (t $\in$ T) in STATES(Ont). The set of all traces over state ontology Ont is denoted by TRACES (Ont). Depending on the application, the time frame T may be dense (e.g., the real numbers), or discrete (e.g., the set of integers or natural numbers or a finite initial segment of the natural numbers), or any other form, as long as it has a linear ordering. The set of *dynamic properties* DYNPROPEXP (Ont) is the set of temporal statements that can be formulated with respect to traces based on the state ontology Ont in the following manner (for an organization or part thereof, Ont is the union of all input, output and internal state ontologies of the roles in the organization (part)).

Given a trace γ over state ontology Ont, the input state of a role at time point t is denoted by state (γ, t, input(r)); analogously, state(γ, t, output(r)), and state(γ, t, internal(r)) denote the output state and internal state of the role. These states can be related to state properties via the formally defined satisfaction relation denoted by the infix predicate $\models$, comparable to the Holds-predicate in the Situation Calculus: state(γ, t, output(r)) $\models$ p denotes that state property p holds in trace γ at time t in the output state of the organism. Based on these statements, dynamic properties can be formulated in a formal manner in a sorted first-order predicate logic with sorts T for time points, Trace for traces and F for state formulae, using quantifiers over time and the usual first-order logical connectives such as $\neg$, &, $\vee$, $\Rightarrow$, $\forall$, $\exists$. Within TTL abstractions can be made by introducing additional terms (e.g., predicates), which are definable in terms of the existing terms.

In order to specify simulation models, a simpler temporal language has been developed, based on TTL. This language (the *leads to* language) enables one to model direct temporal dependencies between two state properties in successive states. This executable format is defined as follows. Let α and β be state properties of the form 'conjunction of atoms or negations of atoms', and e, f, g, h non-negative real numbers. In the *leads to* language $\alpha \twoheadrightarrow_{e,\,f,\,g,\,h} \beta$, means:

If state property α holds for a certain time interval with duration g,
then after some delay (between e and f) state property β will hold for a certain time interval of length h.

A specification of dynamic properties in *leads to* format has as advantages that it is executable and that it can often easily be depicted graphically. Moreover, the language offers the possibility to express both qualitative and quantitative aspects of a process to be simulated. Therefore, it combines the advantages of logic-oriented approaches such as Barringer et al. (1996) and Forbus (1984) with those of mathematical approaches like (Port and Gelder, 1995; Kelso, 1995) in the context of simulation modeling and analysis (Law and Kelton, 2000). For a more precise definition of the *leads to* format, see (Bosse et al., 2005).

Description of relationships between dynamic properties at different levels of aggregation:

(a) provides a basis for formalized biological, cognitive, or organisational theories and their validation,

(b) enables evaluation of trajectories of dynamics (at any level of aggregation) against specified dynamic properties,
(c) allows diagnosis of malfunctioning,
(d) allows relating dynamic properties to empirical physical/chemical/physiological/neurological data.

To aid simulation of processes, languages for specification of high-level executable models can be used, thus obtaining tractable dynamic models for phenomena that, at a too detailed level may be much more complex.

Techniques developed within Artificial Intelligence (and Computer Science) support this methodological perspective. Above all the agent-based modelling paradigm is exploited (see above). Furthermore, for simulation Executable Temporal Logic (Barringer et al., 1996) provides supporting languages and software environments for discrete executable dynamic properties; e.g. (Fisher, 1994; Moss et al., 1998). This can be extended to agent-based continuous-time approaches, e.g., based on real-valued time intervals; (Jonker, Treur, and Wijngaards, 2002) is a contribution in this direction.

The agent-oriented modelling approach can be enriched by recently developed modelling, compositional verification and model checking techniques. Examples of such modelling techniques are process algebra; dynamic and temporal logic; event, situation and fluent calculus; e.g., (Eck et al., 2001; Reiter, 2001). These modelling techniques allow high-level expression of temporal relations, i.e., relations between a state of a process at one point in time, and states at other points in time. In addition, analysis techniques and tools for verification and model checking have progressed to a more mature status in recent years; e.g., Carnegie-Mellon's SMV, Cadence-SMV, and AT&T's SPIN; (Clarke et al., 2000; Manna and Pnueli, 1995; Stirling, 2001).

Compositional verification (Roever, Langmaack, and Pnueli, 1998; Jonker and Treur, 1998; Cornelissen, Jonker, and Treur, 2002) has been developed as a solid method to relate dynamic properties at one aggregation level (e.g., the whole) to dynamic properties at another level (e.g., parts or components), thus enabling formalisation of (domain) theories on dynamics. Also the integration of real-valued variables and continuous time within model checking and verification environments is a useful recent development that can be incorporated; e.g. (Henzinger et al., 1994; Gamboa and Kaufmann, 2001).

3 Meetings formalized

To formalize meetings, its organizational structure, dynamic properties for the overall process, and a protocol for role interactions have to be formalized

3.1 Organizational structure

The American Heritage® Dictionary of the English Language provides the following definitions of a meeting: The act or process or an instance of coming together; an encounter, an assembly or gathering of people, as for a business, social, or religious purpose. We let a meeting be a gathering of people with a common purpose. The common purpose, especially in a business setting, entails that the meeting is convened to discuss issues related to that common purpose, and often with the aim to make some decisions pertinent to the common purpose.

Experience shows that a meeting without some way to structure the discussions takes much longer to reach a conclusion than a meeting in which the discussion is structured. In fact, in any group of more than two people, discussion might lead to different sub-groups discussing the same or different arguments. How can you be sure that everyone hears all the arguments made? How can you be sure that everyone who has something relevant to add to the discussion gets a chance to present her views? How can it be achieved that all people involved (also those who were absent) have access to the results of the meeting afterwards?

Therefore, a common form to structure meetings is the following. A Chairperson chairs every meeting. The Secretary takes minutes of the meeting. Given the chosen meaning of the concept meeting, the dictionary says the following about the Chairperson and Secretary roles:

- Chairperson
  The presiding officer of an assembly, meeting, committee, or board.
- Secretary
  An officer who keeps records, takes minutes of the meetings, and answers correspondence, as for a company.

Depending on the type of meeting, also a Treasurer role may be distinguished; for simplicity this role is not considered in this paper. Taking minutes means writing down the arguments presented by the Participants of the meeting, as well as the decisions made. Chairing a meeting means opening and closing a meeting, making sure that people are talking one at a time, and that only the current issue is being discussed. The decision process differs according to the customs or agreements in the group. Common decision procedures are decision by consensus, decision by majority, and decision by the Chairperson. In graphical form, the general structure of a meeting is as depicted in Fig. 1. A question to be addressed is how dynamic properties describing such a protocol can be identified.

3.2 Organizational behavior

Dynamic properties characterizing an organizational behavior can be specified at different levels: at the level of the organization as a whole (3.2.2), at the level of interactions between roles (interaction protocol) (3.2.3), and at the level of roles. We introduce the topic of organizational behavior by giving a brief overview of pointers to background literature from sociology and organizational management (3.2.1).

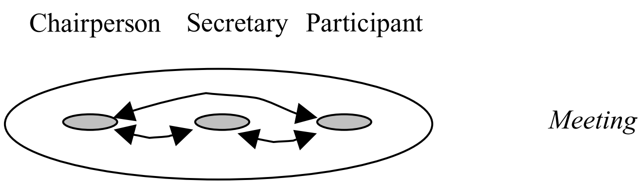


Fig. 1 Generic meeting organization structure

3.2.1 Background

A large body of work exists that investigates the interactions between people in less or more organized forms, of which some also specifically aimed at analyzing the behaviors of persons in and the organization of meetings. We do not intend to give a complete overview of this field, but rather to point out some specific references from the literature. We briefly discuss works of Goffman, (1961, 1963) and Garfinkel (1967). Also, we discuss Robert's rules of order manual (Robert, 2000).

Goffman (1922–1982) studied human interactions, ranging from the way people behave in public to different forms of 'talk'. He was motivated by the differences between 'ourselves' and our social images and developed theories on the idea how people develop identities by means of social actions (e.g., gossip, grunts, etcetera). With respect to the particular topic discussed in our paper (organization of meetings), Goffman has published works on the social organization of gatherings (Goffman, 1961, 1963). The work of Goffman helps us position our research. In the paper, we namely operationalize a rather strict notion of 'meeting', whereas the 'gathering' term of Goffman has a much wider range. For example, interactions may be unfocused ('they result solely by virtue of persons being in one another's presence') or focused ('people effectively agree to sustain for a time a single focus of cognitive and visual attention'). An *encounter* is then a 'focused gathering' and functions as the building block of social organization. A special type of a social organization is a social group. But such a group does not necessarily equate to a gathering or meeting. This becomes clear when the different issues are considered between being a group member and a participant in a gathering, e.g. leadership or tension-management. An aspect of leadership that is extremely important in a gathering is the maintenance of communication ground rules; this does not seem to be as important in group analysis.

Although we may want to say that our use of the term meeting is a special type of gathering in Goffman's terms and therefore rather restricted, it is clear that our notion can be stretched further to include a wider range of 'social interactions'. Still, this paper presents a generic approach for analyzing and studying these types of interactions and therefore not restricted to the meeting example that is worked here.

Ethnomethodology is a perspective on sociology, founded by Garfinkel in the early 1960s (Garfinkel, 1967). Put simply, it studies the way in which people make sense of their social world. It is controversial as it does not assume that this world is orderly–social order is established in the minds of social actors based on a series of impressions and experiences that society confronts an individual with. This field is undoubtedly much larger than the scope of this paper, and this prevents a comparison of the two on that level. Still, ethnomethodology studies social structures relevant for our notion of meetings and our approach of meeting analysis. Garfinkel proposes the *documentary method* that deals with the issue that researchers (e.g., field workers doing ethnographic and linguistic studies) may occasionally not presuppose any knowledge of social structures, but must still eventually decide on facts, hypotheses, conjectures, etcetera. An investigator in a sociological inquiry typically embeds observed events in his presupposed knowledge. But these observations may not be directly witnessed; for example, motivated actions will have to be 'witnessed' over a period of time rather than that is an instantaneous observation. The method involves the search for "... an identical homologous pattern underlying a vast variety of totally different realizations of meaning" (Garfinkel, 1967). In contrast with other (earlier) methods (e.g., those employed by functionalists, Marxists, symbolic interactionists), the underlying pattern is not only derived from individual evidences, but the evidences are also interpreted on the basis of what is known about the underlying pattern. Garfinkel presents an experiment that exaggerates features of the method in order to illustrate it. The study involved a number a students asking a (fake) study counselor questions about some serious problem they needed advice on, where the counselor gave fixed answers without even hearing the questions. Students were asked to comment on their conversation, and these comments were the results of the study giving insights into how students got through the conversation, whether students searched for and perceived a pattern, etcetera. For more details on the experiment, see (Garfinkel, 1967, Chapter 3).

Whereas we saw that our relation to Goffman's work was rather content-based (our strict operational notion of meeting in comparison with his wide definition of gatherings), we see that our work relates to Garfinkel's work on the methodological level. Without making a complete comparison between our approach to meeting analysis and the documentary method (which covers many more social interactions than, again, our strict notion of a meeting), we expect that on this discussion level, the two may benefit from each other. One straightforward example of such benefit is our inclusion of formalization, enabling automated verification and validation, whereas at first sight this cannot be found in Garfinkel's method. We consider future studies in which we elaborate on these methodological differences.

Finally, we discuss Robert's rules of order (Robert, 2000). This is a standard work on guiding people through conducting meetings—how does one carry on business from one meeting to another, who keeps the order, how is a meeting best kept on track, etcetera? Although first published in 1896, it is still considered to be an important guideline and reference for meeting conductance and widely used. It is commonly understood to be a system of parliamentary procedures and functions to promote orderly discourse and debate, to defend the parliamentary rights of the minority, to act on the will of the majority, and to streamline the workings of the business of the organization. There exist some foundations of the rules,[1] for example: no member can speak twice to the same issue until everyone else wishing to speak has spoken to it once; the agenda and all committee reports are merely recommendations—when presented to the assembly and the question is stated, debate begins and changes occur; all remarks must be directed to the Chair. Remarks must be courteous in language and deportment; etcetera. Examples of the rules themselves[1] are: *Point of Privilege*—pertains to noise, personal comfort, etc. —may interrupt only if necessary; *Limit Debate*—closing debate at a certain time, or limiting to a certain period of time; *Reconsider*—can be made only by one on the prevailing side who has changed position or view; etcetera.

Whereas in this paper we have taken other starting points for the initial meeting protocol, and the type of meetings (small, 4 participants, brief) does not necessarily need such heavy machinery as Robert's rules, it is clear that for elaborating on our work and extending our approach (e.g., applying to other types of meetings), consideration of the rules is necessary. Accordingly, we will do so in future extensions of the work described here.

[1] http://www.robertsrules.org

3.2.2 Overall dynamic properties

At the level of the overall organization (which in this case is the meeting as a whole) a number of organization properties can be identified. As an example the following property expresses that no two participants speak at the same time. In this and the following properties, communicates_from_to (*p, q, x, y*) denotes that p communicates to *q* the communicative act *x* with the content *y*. For this paper, we consider two types of communicative acts, mainly inform and declare. Only when the communicative act x is of type declare, then the receiver q is dropped meaning that the message is sent to everyone. For the sake of simplicity, we assume that messages always reach their destination.

OP1
Informal
During the meeting only one Participant is speaking at a time.
Semiformal
At any point in time,
if any participant is speaking,
then all other participants are not speaking
Formal
$\forall$t, p, p$'$:PARTICIPANT, q, q$'$:ROLE, x, x$'$, y, y$'$
p $\neq$ p$'$ & state(γ, t, output(p)) $\models$ communicates_from_to(p, q, x, y) $\Rightarrow$
state(γ, t, output(p$'$)) $\not\models$ communicates_from_to(p$'$, q$'$, x$'$, y$'$)

For an overview of all organization properties specified, see Appendix A. To express the properties of an agenda item, current agenda item and addressed agenda item abstractions are introduced.

Abstraction: agenda item
Informal
An *agenda item* is an item that was declared to be an agenda item and not retracted since then.
Semiformal
Item i is an agenda item if at some point in time it was declared to be so,
and since then it was not declared that it is not an agenda item
Formal
agenda_item_at(γ, i, t) =
$\exists$t$'$ $\leq$ t, m:CHAIR
state(γ, t$'$, output(m)) $\models$ communicates_from_to(m, declare, agenda_item(i))
$\forall$t$''$ t$'$ < t$''$< t => state(γ, t$''$, output(m)) $\not\models$ communicates_from_to(m, declare, not_agenda_item(i))

Abstraction: agenda item
Informal
A *current agenda item* is one that was opened but not yet closed.
Semiformal
An agenda item is a current item if and only if
Some time ago the Chairperson declared that item to be the current item
And since then the Chairperson did not declare the item closed.

Formal
current_agenda_item_at(γ, i, t) =
∀m:CHAIR, t′ ≤ t
state(γ, t′, output(m)) ⊨ communicates_from_to(m, declare, opened(i)) &
∀t″ [t′ < t″ < t ⇒ state(γ, t″, output(m)) ⊭ communicates_from_to(m, declare, closed(i))]

Abstraction: addressed agenda item
Informal
An agenda item has been *addressed* if it was opened and closed during the meeting.
Semiformal
An agenda item has been addressed if and only if for every time point that the chairperson has opened the item, at a later time point she declared the item closed
Formal
addressed_agenda_item_at(γ, i, t) = ∀m:CHAIR
∃t1≤t state(γ, t1, output(m)) ⊨ communicates_from_to(m, declare, opened(i)) &
∀t2≤t state(γ, t2, output(m)) ⊨ communicates_from_to(m, declare, opened(i)) ⇒ ∃t3 t2≤t3≤t &
state(γ, t3, output(m)) ⊨ communicates_from_to(m, declare, closed(i))

3.2.3 Role interaction properties: The generic meeting protocol

A number of role interaction properties have been specified to define a generic interaction protocol for a meeting. For an overview, see Appendix B. Two examples are the following.

RI1 If the Chairperson generates a question (which implies a permission to speak) to a Participant, then a little time later the Participant generates an answer.

Formal
∀m:CHAIR, p:PARTICIPANT ∀t, i
[state(γ, t, output(m)) ⊨ communicates_from_to(m, p, request, i)]
⇒
∃t′ > t, a:act,x [state(γ, t′, output(p)) ⊨ communicates_from_to(p, m, a, x) &
is_in_context(x,i)

RI2 If a Participant requests to add an item to the agenda,
then the Chairperson communicates this to all Participants.
Formal
∀m:CHAIR, p:PARTICIPANT ∀t
state(γ, t, output(p)) ⊨ communicates_from_to(p, m, request, agenda_item(i)))
⇒
∃t′ > t state(γ, t′, output(m)) ⊨ communicates_from_to(m, declare, agenda_item(i)))
Notice that it is easy to add temporal constraints to these properties to denote that the property has to be fulfilled in a predefined number of seconds. For simplicity this has been left out.

4 From protocol to simulation

We describe the main steps how we generate a simulation trace from a given protocol specification.

1. In our terminology, a meeting protocol consists of a number of role interaction properties (RI) that describe the interactions between different roles in a meeting. Some example RI properties are shown above in Section 3.2.3 and a complete list has been included in Appendix B. These properties are basically relationships between events, specifying that if a happens for time duration t, then b happens with time duration t' with some maximal time interval Δ between the intervals t and t'. These properties are deterministic with respect to time duration t and t' (i.e., these do not change during a simulation run); however, stochastic values may be used for time interval Δ. (For the examples included in this paper, the time intervals were also deterministic.)
2. Some initial conditions are given. A simple initial condition in this context may be that the meeting is opened by the chairperson.
3. The protocol specification together with the initial conditions is executed in a logic-based simulation environment (Fasli, 2003; Bosse et al., 2005). The RI properties are implemented as executable temporal rules, forming a kind of temporal rule-base, whereby the initial conditions function as the initial events that start off the simulation.
4. The output of the simulation is a chronological series of events, called a *trace*. A trace describes which properties related to the protocol hold at each time point. Examples of traces can be found below in Tables 1, 2 and 3.

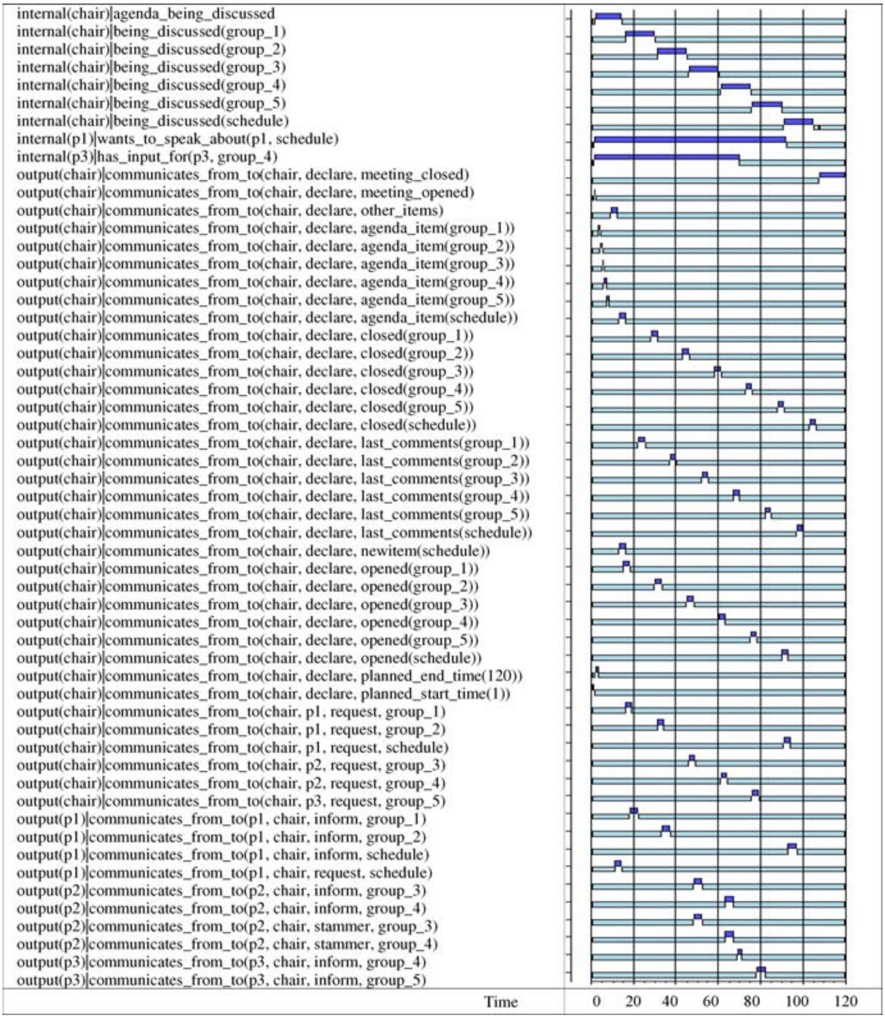
The organization properties (OP) can then be checked against these traces in a verification software environment. For example, one may check if it is indeed true that in some generated trace, no two participants speak at the same time (OP1). We use the verification tool below to check whether some given organization properties hold for some generated trace; based on such an analysis we revise the protocol (i.e., extend the RIs with an exception handling mechanism) such that the properties hold. We iteratively repeat this process until we have a satisfying protocol that agrees with the meetings that we have observed and that meets the organization properties.

It is important to notice that a generated trace thus highly depends on a single protocol specification (and the given initial conditions). However, it is expected that the process of acquiring a ‘good’ protocol will involve enough traces of observed meetings such that a sufficient number of traces out of all possible traces will be covered.

5 Example trace of a simulated meeting based on the generic meeting protocol

The simulations of interest are generated using a logic-based simulation environment. Using this environment, executable temporal rules are specified so that the simulation environment can generate a trace. These executable temporal rules consist of rules that fire based on the current status of the world, without regard to the past. This

Table 1 Simulation trace of a meeting that follows the generic protocol



trace describes which properties related to the protocol hold at each time point. The generated traces can then be analyzed with an automated logic-based checker. This checker takes as input a property of interest about the trace and logically validates the property by the trace. If the property holds in the trace, the checker outputs success otherwise it outputs fail.

We consider a simulation of an example meeting on the topic of study groups. These simulations consist of one chairperson (referred to as chair) and three participants (referred to as p1, p2, and p3). The agenda items are about particular study groups, hence named as group_1, group_2, and so on. For each agenda item one of the participants is the contact person, who is asked to speak if the agenda item is opened.

The simulation discussed here is based on the formal specification of the generic meeting protocol, which was developed based on the meeting guidelines discussed

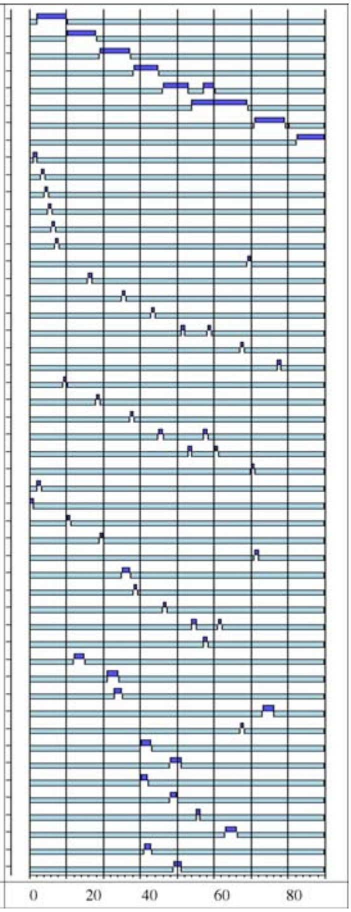
Table 2 The transition from informal statements to formal states

	Informal description	Formal state
0	C: Let's begin in a minute	proposed_begin_time(1)
1	C: Let's try to finish by 5'o clock	communicates_from_to(chair, declare, meeting_opened) proposed_end_time(120)
2	C: We will talk about the regular agenda	communicates_from_to(chair, declare, agenda_item (group_1) communicates_from_to(chair, declare, agenda_item(group_2) communicates_from_to(chair, declare, agenda_item(group_3) communicates_from_to(chair, declare, agenda_item(group_4) communicates_from_to(chair, declare, agenda_item(group_5)
3	C: First, talk about group_1	communicates_from_to(chair, declare, open(group_1))
4	C: Mike, any inputs for group_1	communicates_from_to(chair, p1, request, group_1)
5	Mike gives an explanation on group_1	communicates_from_to(p1, chair, inform, group_1)
6	C: OK, fine.	communicates_from_to(chair, declare, close(group_1))
7	C: Group_2	communicates_from_to(chair, declare, open(group_2))
8	C: Mike, any inputs for group_2	communicates_from_to(chair, p1, request, group_2)
9	Mike gives an explanation on group_2	communicates_from_to(p1, chair, inform, group_2)
10	Mike complains about lecture notes	communicates_from_to(p1, chair, inform, notes)
11	C: This is not the right time for that.	communicates_from_to(chair, p1, revoke, notes)
12	C: Let's move on	communicates_from_to(chair, declare, close(group_2))
13	C: Group_3	communicates_from_to(chair, declare, open(group_3))
14	C: Laurie, any inputs for group_3	communicates_from_to(chair, p2, request, group_3)
15	Laurie explains with frequent stops.	communicates_from_to(p2, chair, inform, group_3) communicates_from_to(p2, chair, stammer, group_3) communicates_from_to(p3, p2, complete, group_3)
16	C: Let's move on	communicates_from_to(chair, declare, close(group_3))
17	C: Group_4	communicates_from_to(chair, declare, open(group_4))
18	C: Laurie, any inputs for group_4	communicates_from_to(chair, p2, request, group_4)
19	Laurie explains with frequent stops.	communicates_from_to(p2, chair, inform, group_4) communicates_from_to(p2, chair, stammer, group_4) communicates_from_to(p3, p2, complete, group_4)
20	C: Let's move on	communicates_from_to(chair, declare, close(group_4))
21	C: Group_5	communicates_from_to(chair, declare, open(group_5))
22	C: Tim, any inputs for group_5	communicates_from_to(chair, p2, request, group_5)
23	Tim speaks more on group_4	communicates_from_to(p3, chair, inform, group_4) communicates_from_to(chair, declare, open(group_4))
24	C: We talked enough on group_4	communicates_from_to(chair, p3, revoke, group_4) communicates_from_to(chair, declare, close(group_4))
25	C: Group_5	communicates_from_to(chair, declare, open(group_5))
26	C: Tim, any inputs for group_5	communicates_from_to(chair, p2, request, group_5)
27	Tim speaks on group_5	communicates_from_to(p3, chair, inform, group_5)
28	C: We are done with the agenda	communicates_from_to(chair, declare, close(group_5))
29	M: I have a comment on schedule	communicates_from_to(p1, chair, request, schedule)
30	C: OK	communicates_from_to(chair, declare, agenda_item (schedule)) communicates_from_to(chair, p1, request, schedule)
31	Mike makes a comment	communicates_from_to(p1, chair, inform, schedule)
32	C: OK, we are done now.	communicates_from_to(chair, declare, close(schedule))
33	C: Same time, next week	communicates_from_to(chair, declare, meeting_closed)

above. The simulation follows the protocol but here we give a brief overview of the trace that is shown in Table 1. The left side of the table lists the predicates used. The right side of the table shows the timeline. A predicate is true over a period of time if there is a bar above the time line; otherwise the predicate is false. The simulation starts by the chairperson declaring the desired end time (proposed_end_time) for the meeting. Next, the chairperson announces the agenda items one by one (agenda_item). Next, the chairperson asks for further additions to the agenda. Participant p1 suggests a new item (schedule), which is also added to the agenda. Once the agenda is finalized, the chairperson opens the first item (group_1) for discussions. The chairperson requests information from the participant who is likely to have input on the current agenda item. After this participant is done speaking, the chairperson asks the other participants to see if they have further information for the topic (last_comments). Since no participant has further input on the agenda item (group_1), the chairperson closes the agenda item and opens the second item. This procedure repeats itself until the agenda item is

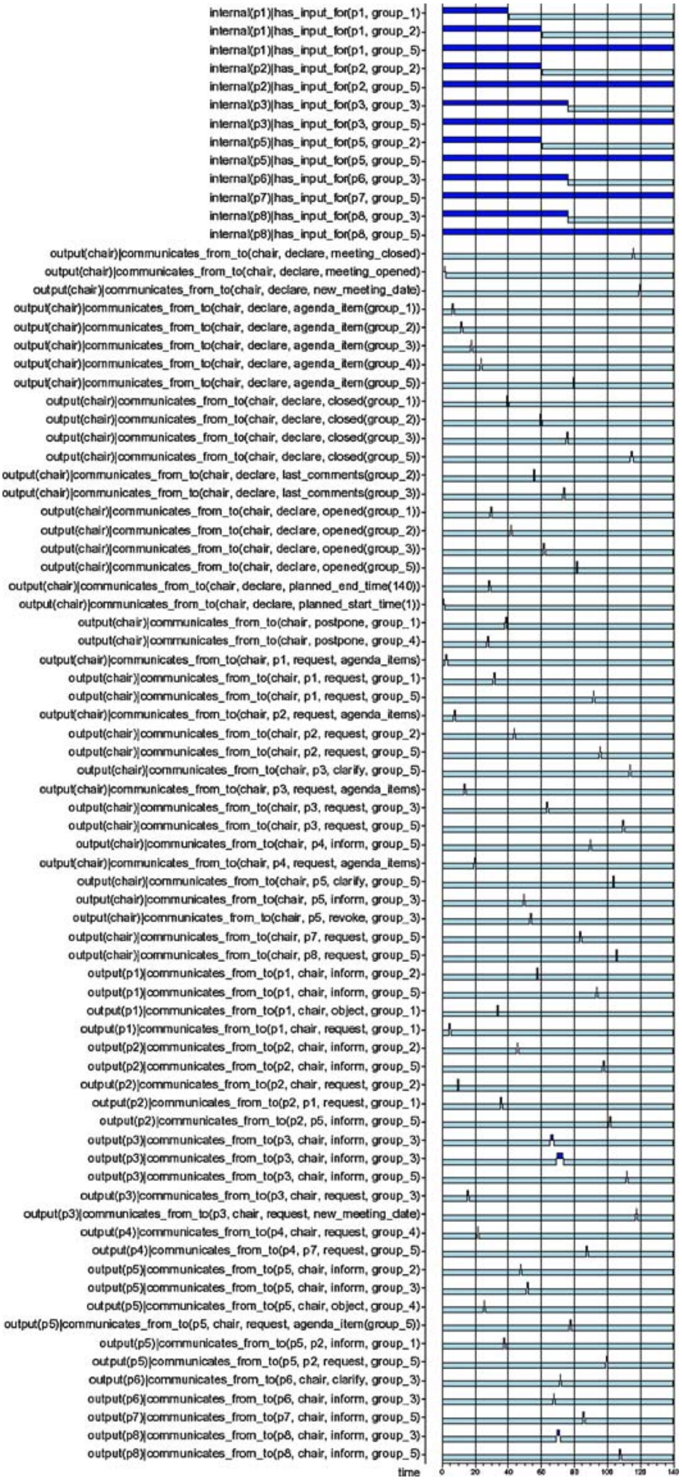
Table 3 Simulation trace based on the refined protocol

internal(chair)	agenda_being_discussed
internal(chair)	being_discussed(group_1)
internal(chair)	being_discussed(group_2)
internal(chair)	being_discussed(group_3)
internal(chair)	being_discussed(group_4)
internal(chair)	being_discussed(group_5)
internal(chair)	being_discussed(schedule)
output(chair)	communicates_from_to(chair, declare, meeting_closed)
output(chair)	communicates_from_to(chair, declare, meeting_opened)
output(chair)	communicates_from_to(chair, declare, agenda_item(group_1))
output(chair)	communicates_from_to(chair, declare, agenda_item(group_2))
output(chair)	communicates_from_to(chair, declare, agenda_item(group_3))
output(chair)	communicates_from_to(chair, declare, agenda_item(group_4))
output(chair)	communicates_from_to(chair, declare, agenda_item(group_5))
output(chair)	communicates_from_to(chair, declare, agenda_item(schedule))
output(chair)	communicates_from_to(chair, declare, closed(group_1))
output(chair)	communicates_from_to(chair, declare, closed(group_2))
output(chair)	communicates_from_to(chair, declare, closed(group_3))
output(chair)	communicates_from_to(chair, declare, closed(group_4))
output(chair)	communicates_from_to(chair, declare, closed(group_5))
output(chair)	communicates_from_to(chair, declare, closed(schedule))
output(chair)	communicates_from_to(chair, declare, opened(group_1))
output(chair)	communicates_from_to(chair, declare, opened(group_2))
output(chair)	communicates_from_to(chair, declare, opened(group_3))
output(chair)	communicates_from_to(chair, declare, opened(group_4))
output(chair)	communicates_from_to(chair, declare, opened(group_5))
output(chair)	communicates_from_to(chair, declare, opened(schedule))
output(chair)	communicates_from_to(chair, declare, planned_end_time(120))
output(chair)	communicates_from_to(chair, declare, planned_start_time(1))
output(chair)	communicates_from_to(chair, p1, request, group_1)
output(chair)	communicates_from_to(chair, p1, request, group_2)
output(chair)	communicates_from_to(chair, p1, request, schedule)
output(chair)	communicates_from_to(chair, p1, revoke, notes)
output(chair)	communicates_from_to(chair, p2, request, group_3)
output(chair)	communicates_from_to(chair, p2, request, group_4)
output(chair)	communicates_from_to(chair, p3, request, group_5)
output(chair)	communicates_from_to(chair, p3, revoke, group_4)
output(p1)	communicates_from_to(p1, chair, inform, group_1)
output(p1)	communicates_from_to(p1, chair, inform, group_2)
output(p1)	communicates_from_to(p1, chair, inform, notes)
output(p1)	communicates_from_to(p1, chair, inform, schedule)
output(p1)	communicates_from_to(p1, chair, request, schedule)
output(p2)	communicates_from_to(p2, chair, inform, group_3)
output(p2)	communicates_from_to(p2, chair, inform, group_4)
output(p2)	communicates_from_to(p2, chair, stammer, group_3)
output(p2)	communicates_from_to(p2, chair, stammer, group_4)
output(p3)	communicates_from_to(p3, chair, inform, group_4)
output(p3)	communicates_from_to(p3, chair, inform, group_5)
output(p3)	communicates_from_to(p3, p2, complete, group_3)
output(p3)	communicates_from_to(p3, p2, complete, group_4)
	time



(*Continued on next page*)

Table 3 (*Continued*)



group_4. On this agenda item, when the chairperson asks for other comments from the participants, participant p3 provides additional comments. Later the meeting continues as before. After the last agenda item is discussed, the chairperson declares that the meeting is closed.

From a broad overview, the simulation described above has some differences from our observations of real meetings. For this reason, we observed a real meeting and obtained data on how it was carried out. These data were analyzed in some depth.

6 An empirical trace of a real meeting

An important part of the work presented here is based on empirical data. This data was obtained through carefully observing a meeting in the Artificial Intelligence Department of the Vrije Universiteit Amsterdam. Similar to the observation techniques explained elsewhere (Serman and Basili, 1998), the observer sat apart from the meeting participants and the chair. Two of the participants and the chair knew why the observant was present, while a third participant did not.

The observer wrote down the conversations of the meeting in an informal language. Later these informal texts were formalized to analyze and reason about the meeting. Table 2 gives an overview of these conversations. The left column in the table provides the informal text and the right column gives the formalized states.

We briefly explain the differences from the simulated meeting trace in Section 5. The trace again starts with the chairperson announcing a desired end time for the meeting (`proposed_end_time`). The chairperson announces the agenda items but does not explicitly ask for additions to the agenda. After the chair opens an agenda item and receives input on the item, she closes the item when she sees fit. Compared to the generic meeting protocol described in Section 3.3.2, the difference here is that the chair does not explicitly ask for further input from the participants. Complementing this is a change in the role behavior of participants. Whereas in the meeting simulated according to the generic protocol (Section 5), a participant speaks only when permission is given, in the real meeting participants take the initiative to speak up without being asked. The interesting question then is how these different behaviors affect the outcome of the meetings? Do the desired properties of interest hold for both cases? Does one trace have advantages over the other one? We discuss these questions next.

7 Formal analysis of simulated trace and empirical trace

We analyze the traces generated by these simulations in terms of the organization properties defined above. To do so, the first eight organization properties (Appendix A) have been entered into the checker and automatically checked against each trace.

7.1 Analysis of the simulated meeting

The meeting simulated according to the generic protocol (Section 5) satisfies the first organization property (OP1), which states that no two participants speak at the same time. This is intuitive since participants speak only when given permission. In this simulation, the chair ensures that only one participant has the permission to speak. Hence, the property holds. The second property (OP2) is on the agenda items that were talked. The role interaction RI10 specifies that once an agenda item is closed, the chair chooses a new item from the agenda. Hence, it is always the case that the chairperson will open an existing agenda item. This explains why OP2 holds for this trace as well.

OP3 is satisfied for this trace because before closing each topic the chairperson asks for further comments from the participants. Hence, anyone who declares an intention to speak gets a chance to speak. Organization property OP4 states that the meeting is eventually closed. This will always hold for a meeting based on the generic meeting protocol as long as the number of items on the agenda as well as the duration of comments on the items are finite. OP5 ensures that no meeting ends prematurely; that is if the meeting ends, then all agenda items have been discussed. In the specification of the meeting, the only way to close a meeting is when the meeting items have been discussed. OP6 states that no two items are open at the same time. This holds for this trace again due to role interaction RI10. A chairperson will open a new agenda item only if the previous item is closed. Organization property OP7 states that if a participant is speaking then she is speaking on the current item. This follows from the fact that the chairperson will only allow a participant to speak on the current item (RI3). Organization property (OP8) states that the meeting starts and ends on time. This property holds for this trace since the meeting starts with declarations of the start and end times of the meeting and the meeting takes place between these time points. However, in general this property may have conflicts with OP3.

7.2 Analysis of the empirical data of the real meeting

While the generic meeting protocol obeys the organization properties, the real meeting trace violates some of them. To avoid repetition, only the properties that are violated are discussed here.

The first interesting situation happens during the discussion of item group_3 (see lines 13–16). The chairperson requests information from p2 on the item. The participant p2 speaks with short breaks (`stammer`), which influences one of the other participants (p3) to help p2 with his speech (`complete`). Notice that this is not part of the generic protocol and in general no participant has to help other participants. To be able to generate this behavior, we added an extra role interaction property to the simulation so that participant p3 would help p2. Participant p3's helping p2 is constructive in that it allows p2 to formulate his thoughts. Ironically, this situation disobeys one of the desired organization properties of meetings; namely OP1 which states that no two participants at a meeting should speak at the same time.

After chairperson requests information from a participant, the participant provides the required information. In some cases, it could also be the case that the participant provides information that is not relevant to the request of the chairperson. One such example happens during the discussion of item group_2 (see lines 7–12). After giving feedback on group_2, participant p1 starts speaking on a topic (`notes`) that is out of the scope of group_2. This is an example of impromptu interruption from participants that sometimes happens. This behavior of p1 causes the violation of the organization property OP7, which says that participants speak on current agenda items only. While this behavior of the participant is not part of the generic interaction protocol, a method for recovering from such a situation is followed in the meeting. Hence, the chairperson can first revoke the permission from participant p1 and then continue with the protocol.

Contrary to the generic protocol, in this simulation the chairperson does not request further input from other participants before closing an agenda item. One interesting consequence is that after the discussion of item group_4, the chairperson closes the agenda item (line 20). However, there is still a participant who is willing to speak more on the item. Hence, this participant (participant p3) continues speaking about group_4, even though the item has been closed and a new item has been opened (line 23). This point in time is interesting because in reality both agenda items are current. Item group_5 is current because it has been declared as open and not closed by the chairperson. However, group_4 is also current, since one participant is talking about this item. Hence, another organization property, property OP6 is violated since there are two current items at the same time. However, this failing of this property does not halt the system. The meeting handles this exception in the sense that the chairperson in this case lets the participant finish and then re-closes the item group_4 and reopens the item group_5 (in lines 24 and 25).

8 Refined protocol and simulation

As shown in the analysis in Section 7, a real meeting (such as the one described in Section 6) may deviate from a meeting correctly following the protocol (such as the simulated meeting in Section 5) in the following ways:

- sometimes, by exception, protocol properties are violated by one of the members
- strategies are employed to handle these exceptions and to put the meeting on the right track again

One of the reasons that these exceptions occur are the fact that human agents are not ideal and may forget things. In practice members are able to accept these shortcomings and to recover from them. To this end a number of exception handling strategies are used. This can be considered a more sophisticated way of working than just by following the protocol. An interesting question is whether the generic meeting protocol can be refined by including such exception handling strategies to provide a more robust protocol. This question is discussed in the current section.

To experiment with a refined protocol, using the formal states given for the empirical trace, a second simulation was developed, where a number of the rules for the simulation (as used in Section 5) were adapted to reconstruct the empirical trace as precisely as possible. The generated trace indeed closely resembles our observations of the real meeting described in Section 6. For example, the exception of the participant speaking on notes while the current agenda item is group_2, is now handled realistically in the simulation: the chairperson first revokes the permission from participant p1 and then continues with the protocol. Moreover, now the simulated meeting can also handle the exception such as a participant speaking on an already closed agenda item i2 when the current agenda item is i1. By the addition of exception handling strategy the chairperson returns to the earlier agenda item i2, lets the participant finish and then recloses the item i2 and reopens the item i1. The following rules, which can be considered part of such a refined protocol, were used to obtain this. These two rules together constitute RI6 in Appendix B.

RI-Make current

If after a new agenda item was opened and not yet closed, a Participant speaks on an earlier addressed agenda item, then the Chairperson closes the current agenda item and reopens the earlier item.

Formal

∀t, i1,i2 ∀m:CHAIR, p:PARTICIPANT ∀y
[current_agenda_item_at((γ, i2, t) &
addressed_agenda_item_at(γ, i1, t) &
state(γ, t, output(p)) $\models$ communicates_from_to(p, m, inform, y) & in_context_of(i1)]
$\Rightarrow$ ∃t$''$ $\geq$ t state(γ, t$''$, output(m)) $\models$ communicates_from_to(m, declare, closed(i2)) &
state(γ, t$''$, output(m)) $\models$ communicates_from_to(m, declare, opened(i1))

RI-Revoke if necessary

If a Participant speaks on an item other than the current agenda item or any earlier addressed agenda item, then the Chairperson revokes the Participant and asks for additional comments on the current agenda item from the other participants.

Formal

∀t, i2 ∀m:CHAIR, p:PARTICIPANT, y
[state(γ, t, output(p)) $\models$ communicates_from_to(p, m, inform, y) &
not ∃i1 addressed_agenda_item_at(γ, i1, t) & in_context_of(y, i1)]
$\Rightarrow$ ∃ t$''$$\geq$t state($\gamma$, t$''$, output(m)) $\models$ communicates_from_to(m, p, revoke, i1) &
∀q state(γ, t$''$, output(m)) $\models$ communicates_from_to(m, q, request, info_on(i1))

9 Analysis of the organizational properties for the third trace

Once, we refined the protocol we now need to verify how this protocol handles another real meeting. For this reason, we examine another real meeting. The trace for this meeting is given above. The organization properties that were mentioned above are also checked against this third trace. We discuss the analysis of the trace in relation to these organization properties. OP1 fails for this trace, as it did for the previous trace. Sometimes it is difficult to ensure that participants do not speak at the same time. For this reason, we relax this organization property into a new OP1-weak property, which states that no two participants speak at the same time for more than five seconds. The trace satisfies this new property. The OP2 property holds for the trace. That is, if the chair officially opens up a discussion, this is always one of the agenda items. OP3 fails for the trace. One of the participants (p1 in the trace) requests item group_1. Later, when the item is opened for discussion, objects to the item as irrelevant for the current meeting and requests it to be scheduled for a later meeting. Even though this participant has input for the item, the meeting ends without him expressing his input. OP4 is satisfied for the trace, since the meeting is eventually closed. OP5 fails for this trace. OP5 states that all agenda items are addressed by discussing them. However, in this trace, an agenda item is objected by a participant, which causes the item to be removed from the agenda. In this case, the item has been postponed to a later meeting. To handle this situation, we propose a weaker version of OP5, OP5-weak, which states that an agenda item is addressed if it has been discussed or if it has been postponed to another meeting. OP6 holds for this meeting. Putting an additional time limit ensures that short conflicts do not interrupt the flow of the meeting as long as the meeting can continue its operation as before. OP7 states that the meetings start and end on time, which holds for this trace.

10 Analysis of role interactions for the second and the third traces

Appendix B gives a list of role interactions for our meeting protocol. The interactions that are marked as RI are those originally thought to be the role interactions. However, as new traces were generated, some of these proposed role interactions were no longer sufficient to generate the interactions that took place in the traces. Sometimes, a role interaction was too restrictive and sometimes a new role interaction was needed to handle cases that were not thought of initially. For these reasons, the role interactions were coupled with more interactions that are labeled as exception handling interactions. These exception-handling interactions are meant to be used only if the main role interaction fails for a reason. This section analyzes how the traces accommodate these role interactions.

Both traces satisfy RI1 since when the chairperson requests some information on a topic from a particular participant, the participant always provides some information. In Trace 2, the supplied information is always in the type of an `inform`, meaning that the participant gives information on the item. For Trace 3, sometimes the provided information is meta-level in that the participants do no talk on the item but on how to process the item; such as suggesting to postpone an item. However, in both traces there is always an answer from the participant. Even though this is the case, it is not hard to envision a meeting where a participant does not respond to a question. For such cases, an interaction property for exception handling is added. This new interaction property allows a chairperson to direct her query to a second participant if the first participant does not speak within a certain time frame.

Both traces satisfy RI2. If a participant requests an agenda item, then the item is added and communicated to all participants. This interaction property holds because no one objects to agenda item requests before they are added. If this had not been the case, RI2 would have failed and we would need an exception handling property to handle the recovery. Whereas no objections take place before an item is added to the agenda, in Trace 3 an objection takes place after an agenda item is added to the agenda. To deal with such cases, RI3 is added to the protocol. This interaction states that if a participant objects to an agenda item, then the chairperson has the freedom to choose between rejecting the objection or accepting the objection and postponing the item.

In the initial protocol, the role interaction property RI4 stated that if a participant is speaking, then nobody else should be speaking. However, this is violated in Trace 2 as was extensively described in the discussion of OP1. For this reason, the interaction property is weakened so that it now states that no two participants speak at the same time for a small time frame. This weakened role interaction property is also satisfied in Trace 3.

Trace 2 violates RI5, which states that participants only speak on current agenda items. This requirement is violated because at one point in the trace, a participant continues on speaking on an item that is closed. To handle this exception, we add a role interaction property that states that if the item does not become current (by the chairperson closing the current and opening again the previous item), then the participant will stop speaking in a certain time frame. This weaker requirement is also satisfied by Trace 3.

RI7 states that if a participant is revoked permission to speak, then she will do so in a short time frame. This is satisfied by both traces. In certain traces, the chairperson might end up revoking the permission more than once to make the participant stop speaking. R10 states that if no participant has further input on an item, then the chairperson closes the item. This is satisfied in both traces. Similarly, R11 states that if a participant has more input on the item, then the chairperson will open a new round for last comments. This is also satisfied in both traces. Note that this is not the only way to close an item, but a fair way to do so. That is, an agenda item can be closed without asking for further input from the participants. However, if R10 and R11 are forced to be the only ways to close an item, then a fairness property is enforced. This way anyone, who has something to say on an agenda item, gets a chance to do so.

R10 implies that all agenda items in the meeting will be discussed. This property is satisfied by Trace 2. However, the interaction property has to be revised for Trace 3, since in Trace 3 an agenda item is postponed to another meeting. The revision suggests that all agenda items are treated in a meeting, either by discussing it or by postponing it to another meeting. The exception handling interaction property again serves this purpose. The weaker intersection property states that if some agenda items are not opened for discussion, then they must have been postponed to another meeting. This exception can now be handled successfully. RI11 holds for both traces. The chairperson announces an end-time for the meeting during the meeting. Similarly, after the proposed end-time, the chairperson opens an item. Thus, RI12 also holds for both traces. RI13 holds for Trace 2. The chair first proposed an end-time and then lists the items. However, R13 fails for Trace 3 since the chairperson announces an end-time based on the agenda (i.e., after stating agenda items). For this reason, another property is added. This property weakens the temporal dependency and states that the chairperson announces the agenda items at some point during the meeting. RI15 is satisfied by both traces and both meetings are finally closed.

11 Discussion

This paper formalizes a generic role interaction protocol for meetings using the logical language TTL; cf. (Jonker and Treur, 2002). This protocol adheres to several meeting guidelines. In addition, using TTL, desirable properties for meetings can be formally represented. An important advantage of formalizing the protocol with an executable subset of TTL is that the protocol can be simulated. The simulation yields a trace in which properties of protocols can be checked. Next, this simulated trace is compared to an emprical trace of a meeting. Based on deviations revealed in this comparison, we refine the existing protocol to capture more human-like interactions so that the protocol can be simulated to produce a trace that is identical to that of the emprical trace. Ultimately, we aim to have a simulation that captures the processes underlying how meetings are carried out and use these processes as (formal) meeting protocols.

Croston and Goulding present one of the earlier empirical works on meeting effectiveness (1966). Croston and Goulding develop a meeting analysis kit that is used in different departments of a company by the participants of the meeting. The kit enables the participants to reevaluate a past meeting by analyzing the topics discussed, the time spent on each topic, and so on. Based on the analysis from different meetings, Croston and Goulding observe that the starting a meeting with a formal agenda and better chairing of the meetings increase the effectiveness of meetings. The meeting protocol that we propose respects both of these observations. Further, we explicitly formalize the notion of better chairing a meeting.

Serman and Basili study various properties of software inspection meetings in a software development project (1998). Similar to the generation of the empirical trace here, Serman and Basili collect data by attending inspection meetings as an observant. They later analyze their data statistically to uncover causal relations between various properties of the meeting, such as effectiveness, efficiency, or meeting length. While Serman and Basili discover interesting relations, they do not provide a formal protocol of how the meetings should be carried out as we have done here. Since our study uses simulations, we can easily adjust different behaviors of participants to see the effect of (local) properties of participants of a meeting on the (global) properties of the meeting as a whole.

Generally, the group-support systems help participants share data, improve communication, and reach decisions. Hence, group-support systems can help increase the efficiency of meetings. Niederman et al. study the meetings in organizations with group-support systems (1996). Their primary focus is to show how the use of group-support systems by facilitators affects meeting performances. Through interviews with facilitators, Niederman et al. observe that different facilitators have different ideas on measuring performance. However, no formal rules for identifying or bringing out successful meetings are identified.

Given the informal literature as discussed, the work reported in the current paper contributes some first steps in formal analysis of meetings. We show that the meeting protocols that adhere to the guidelines rigidly do not necessarily resemble human meetings, which exploit more sophisticated strategies. We study how this discrepancy can be overcome by including exception handling strategies within the protocol. Future research will address this theme further.

Appendix A: Overall organizational behavior properties

At the level of the overall organization (which in this case is the group as a whole) a number of organization properties have been identified.

OP1: No speaking at the same time

Informal
During the meeting only one Participant is speaking at a time.
Semiformal
At any point in time,
if any participant is speaking,
then all other participants are not speaking
Formal
$\forall$t, p, p$'$:PARTICIPANT, q, q$'$:ROLE, x, x$'$, y, y$'$
p $\neq$ p$'$ & state(γ, t, output(p)) $\models$ communicates_from_to(p, q, x, y) $\Rightarrow$
state(γ, t, output(p$'$)) $\not\models$ communicates_from_to(p$'$, q$'$, x$'$, y$'$)

OP1-weak: No speaking at the same time for three seconds

Abstraction:
Informal
A participant spoke for consecutive s seconds.
Semiformal
A participant p spoke for s seconds if at some point in time she started speaking,
And since then there has not been any time when she was not speaking.
Formal
spoke(γ, p, t, s) =
$\forall$q, q$'$:ROLE, x, x$'$:ACT, y, y$'$ m: CHAIR
$\exists$t$'$ $\leq$ t
state(γ, t$'$, output(m)) $\models$ communicates_from_to(p, q, x, y) & (t-t$'$)$\geq$s &
$\forall$t$''$ (t$'$ < t$''$ $\leq$ t) $\Rightarrow$ state(γ, t$''$, output(m)) $\models$ communicates_from_to(p, q$'$, x$'$, y$'$)

Informal
During the meeting only one Participant is speaking at the same time for a duration of 3 seconds
Semiformal
At any point in time,
if any participant spoke for 3 seconds,
then none of the other participants spoke for the same 3 seconds.
Formal
$\forall$t, p, p$'$:PARTICIPANT [(spoke(γ, p, t, 3) & spoke(γ, p$'$, t, 3)) $\Rightarrow$ p = p$'$]

OP2
Informal
During the meeting only agenda items are addressed.
Semiformal
At any point in time t,
if item i is opened
then i is an agenda item
Formal

$\forall$t, i, p, q, x, y $\forall$m:CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, opened(i))
$\Rightarrow$ agenda_item_at(γ, i, t)

OP3
Informal
Every Participant who indicates that he has something to say on the current agenda item will have the opportunity to speak.
Semiformal
At any point in time t,
if at time t a participant communicates that he has something to say about the current agenda item i
then before the item was closed a later time point t′ exists such that at t′ the participant communicates something in the context of i
Formal
$\forall$t, t2, I, p:PARTICIPANT, q:ROLE, m: CHAIR
current_agenda_item_at(γ, i, t) &
state(γ, t2, output(m)) $\models$ communicates_from_to(m, declare, closed(i)) =>
state(γ, t, output(p)) $\models$ has_input_for(p, i) $\Rightarrow$
$\exists$t′ $\leq$ t < t2, x
state(γ, t′, output(p)) $\models$ communicates_from_to(p, q, inform,x)
& is_in_context_of(x, i)

The notion of being in context of is assumed a given notion.

OP4
Informal
Eventually the meeting is closed.
Semiformal
At some point in time the chairperson declares the meeting closed
Formal
$\forall$m:CHAIR $\exists$t state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed)

OP5
Informal
If the meeting is closed, all agenda items have been addressed.
Semiformal
At any point in time,
if the meeting is declared closed,
then for any item i that was on the agenda there are earlier time points at which item i was declared opened and closed
Formal
$\forall$t, i, m:CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed) &
agenda_item_at(γ, i, t) $\Rightarrow$
addressed_agenda_item_at(γ, i, t)

OP5-weak
Informal
If the meeting is closed, all agenda items have been addressed.
Semiformal
At any point in time,
if the meeting is declared closed,
then for any item i that was on the agenda there are earlier time points at which item i was declared opened and closed or there is an earlier time point at which the item is postponed to another meeting.
Formal
∀t, i, m:CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed) & agenda_item_at(γ, i, t) $\Rightarrow$
addressed_agenda_item_at(γ, i, t) OR $\exists t' \leq t$ state(γ, t, output(m)) $\models$ communicates_from_to(m, postpone, agenda_item(i))

OP6
Informal
No two items are current.
Semiformal
At any point in time t,
if item i is current at t,
and item i′ is current at t,
then $i = i'$
Formal
∀t, i, i′
current_agenda_item_at(γ, i, t) & current_agenda_item_at(γ, i′, t) $\Rightarrow$ $i = i'$

OP7
Informal
The meeting starts and ends in time.
Semiformal
The meeting starts at the planned starting time and ends before the planned end time
Formal
∀m:CHAIR ∀t1 [planned_starting_time(t1) $\Rightarrow$
state(γ, t1, output(m)) $\models$ communicates_from_to(m, declare, meeting_opened)] &
∀t2, t3 state(γ, t2, output(m)) $\models$ communicates_from_to(m, declare, planned_end_time(t3)) $\Rightarrow$
$\exists t4 \leq t3$ state(γ, t4, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed)

OP8
Informal
Every communication in the meeting is received by everyone
Semiformal
At any point in time,
if a participant communicates something to another one,
then this communication will be received by everyone

Formal
$\forall$t, p, q , q′:ROLE, x, y
state(γ, t, output(p)) $\models$ communicates_from_to(p, q, x, y) $\Rightarrow$
$\exists$t' $\geq$t state(γ, t, input(q′)) $\models$ communicates_from_to(p, q′, x, y)

OP9
Informal
The secretary will make minutes of the meeting
Semiformal
if an agenda item is closed,
then notes for the minutes on this item have been made by the Secretary
Formal
$\forall$t, i $\forall$m:CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, closed(i)) $\Rightarrow$
state(γ, t, EW) $\models$ notes_present_for_by(i, Secretary)

OP10
Informal
The internal state property of the chairperson indicating that i is being discussed holds precisely then when i is a current agenda item
Semiformal
i is a current agenda item,
iff the internal state property of the chairperson that i is being discussed holds
Formal
$\forall$t, i $\forall$m:CHAIR
current_agenda_item_at(γ, i, t) $\Leftrightarrow$
state(γ, t, internal(m)) $\models$ being_discussed(i)

Appendix B: Role interaction properties: The generic meeting protocol

The following role interaction properties define an interaction protocol for the meeting.

RI1 If the Chairperson generates a question (which implies a permission to speak) to a Participant on an agenda item, then a little time later the Participant will start speaking on the agenda item.
Formal
$\forall$m:CHAIR, p:PARTICIPANT $\forall$t, i
[state(γ, t, output(m)) $\models$ communicates_from_to(m, p, request, i))]
$\Rightarrow$
$\exists$t′ > t, a:ACT, x [state(γ, t′, output(p)) $\models$ communicates_from_to(p, m, a, x)
is_in_context_of(x,i)]
Exception Handling:
If the Participant does not generate an answer after being requested to speak within 5 seconds, Then the Chairperson requests the answer from another participant that has not been directed the same question.
$\forall$m:CHAIR, p:PARTICIPANT $\forall$t, q
[state(γ, t, output(m)) $\models$ communicates_from_to(m, p, request, q)) &
not $\exists$t′ > t, a,b:act [state(γ, t′, output(p)) $\models$ communicates_from_to(p, m, b,

answer_on(a, q))) & $(t' - t) < 5$]
$\Rightarrow$
$\exists t'' > t$, r:PARTICIPANT [state(γ, t'', output(m)) $\models$ communicates_from_to(m, r, request, q) & $p \neq r$]

RI2 If a Participant requests to add an item to the agenda, then the Chairperson communicates this to all Participants.
Formal
$\forall$m:CHAIR, p:PARTICIPANT $\forall$t, i
state(γ, t, output(p)) $\models$ communicates_from_to(p, m, request, agenda_item(i))
$\Rightarrow$
$\exists t' > t$ state(γ, t', output(m)) $\models$ communicates_from_to(m, declare, agenda_item(i))

RI3 If a participant objects the addition of an agenda item, Then the Chairperson postpones the agenda item or makes it an agenda item.

$\forall$m:CHAIR, p:PARTICIPANT $\forall$t, i
state(γ, t, output(p)) $\models$ communicates_from_to(p, m, object, agenda_item(i))
$\Rightarrow$
$\exists t' > t$ [state(γ, t', output(m)) $\models$ communicates_from_to(m, postpone, agenda_item(i))
OR
state(γ, t', output(m)) $\models$ communicates_from_to(m, declare, agenda_item(i))

RI4 If a participant is speaking, then nobody else is speaking.
Semiformal
At any point in time t,
if p is communicating some information,
then no other participant is communicating information.
Formal
$\forall$t, i, p, q :ROLE, x, y, x', y'
state(γ, t, output(p)) $\models$ communicates_from_to(p, q, x, y)
$\Rightarrow$ $\forall r \neq p$, s :ROLE state(γ, t, output(r)) $|\neq$ communicates_from_to(r, s, x', y')

Exception Handling:
If more participants are speaking at the same time, then this happens for at most 5 seconds.

$\forall$m:CHAIR, p,q:PARTICIPANT $\forall t,t',t'',t'''$
[speaks_from_until(p, t,t') &
speaks_from_until(q,t'',t''') &
$p \neq q$]
$\Rightarrow$
$(t' - t'') < 5$

RI5 If a participant is speaking, then she is speaking on the current item.
Semiformal

At any point in time t,
if at t the item i is current agenda item
and at t any participant is communicating X,
then X fits in item i

Formal

$\forall$t, i, p, q :ROLE, x, y
current_agenda_item_at(γ, i, t) &
state(γ, t, output(p)) $\models$ communicates_from_to(p, q, x, y)
$\Rightarrow$is_in_context(y, i)

RI6 If the Participant speaks on an agenda item that is not the current one, then she will stop speaking in 5 seconds.

$\forall$m:CHAIR, p:PARTICIPANT $\forall$t, x, i
[state(γ, t, output(p)) $\models$ communicates_from_to(p, m, inform, x) &
in_context(x,i) &
not current_agenda_item(i, t)]
$\Rightarrow$
not spoke(γ, p, t + 5, 5)

Exception Handling:

If the Participant speaks on an agenda item that does not become current during her talk
And the Participant does not stop speaking in 5 seconds
Then the chairperson will either make the agenda item current or revoke the permission of the Participant to speak.

$\forall$m:CHAIR, p:PARTICIPANT $\forall$t, x, i
[state(γ, t, output(p)) $\models$ communicates_from_to(p, m, inform, x) & in_context(x,i) &
not current_agenda_item(i, t) &
$\forall$t$'$ > t (t$'$−t)<5 $\Rightarrow$[not current_agenda_item(i, t$'$) & state(γ, t$'$, output(p)) $\models$ communicates_from_to(p, m, inform, x)]
$\Rightarrow$
$\exists$t$'''$ > t$'$ [current_agenda_item(I, t$'''$) OR state(γ, t$'''$, output(m)) $\models$ communicates_from_to(m, p, revoke, i)]

RI7 If the Chairperson revokes the permission to speak from a Participant while that Participant is still speaking, then that Participant will stop speaking in 2 seconds.

Formal

$\forall$t, i $\forall$m:CHAIR, p:PARTICIPANT $\forall$ x, y
state(γ, t, output(m)) $\models$ communicates_from_to(m, p, revoke, i)
& state(γ, t, output(p)) $\models$ communicates_from_to(p, x, y, i)
$\Rightarrow$ $\forall$x$'$, y$'$, z$'$ $\exists$t$'$ [0 $\leq$ (t$'$- t) $\leq$ 2, state(γ, t$'$, output(p)) $\not\models$ communicates_from_to(p, x$'$, y$'$, z$'$)]

Exception Handling:
If the Chairperson revokes a permission of a Participant after which the participant has not stopped speaking,
Then the Chairperson will revoke the permission of the Participant again.
$\forall$t, i $\forall$m:CHAIR, p:PARTICIPANT
state(γ, t, output(m)) $\models$ communicates_from_to(m, p, revoke, i) &
state(γ, t, output(p)) $\models$ communicates_from_to(p, x, y, z) &
$\exists$x$'$, y$'$, z$'$ state(γ, t+1, output(p)) $\models$ communicates_from_to(p, x$'$, y$'$, z$'$)
$\Rightarrow$
$\exists$ t$'$ $\geq$ t+1, state(γ, t$'$, output(m)) $\models$ communicates_from_to(m, p, revoke, i)

RI8 If the chairperson has asked for last comments and for 5 seconds after that, no participant having information regarding the current item, puts forward their information, and no participant raises its hand to put forward information, then the Chairperson closes the agenda item.
Formal
$\forall$t, i $\forall$m:CHAIR
[$\exists$t$'$ $\leq$ t – 5, state(γ, t$'$, output(m)) $\models$ communicates_from_to(m, request, last_comments(i)) &
[$\exists\neg$p:PARTICIPANT, $\exists\neg$t$''$, t$'$ $\leq$ t$''$ $\leq$ t &
[state(γ, t$''$, output(p)) $\models$ communicates_from_to(p, m, inform, i) OR
state(γ, t$''$, output(p)) $\models$ raises_hand(p,i) OR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, closed(ag:agenda_item))]]]
$\Rightarrow$
$\exists$t$'$ $\geq$ t state(γ, t$'$, output(m)) $\models$ communicates_from_to(m, declare, closed (ag:agenda_item))

RI9 If all information on the current agenda item has been exchanged and for 5 seconds no participant has put forward information or raised his hand, then the chairperson asks for last comments.
Formal
$\forall$t,t$'$, i $\forall$m:CHAIR
[last_communication_at_before(γ, t, t$'$) &
$\exists\neg$p:PARTICIPANT, $\exists\neg$t$''$, (t$'$ – 5) $\leq$ t$''$ < t &
[state(γ, t$''$, output(p)) $\models$ communicates_from_to(p, m, inform, i) OR
state(γ, t$''$, output(p)) $\models$ raises_hand(p,i)] &
state(γ, t, output(m)) $\not\models$ communicates_from_to(m, declare, last_comments (ag:agenda_item))]
$\Rightarrow$ $\exists$t$'''$ $\geq$ t state(γ, t$'''$, output(m)) $\models$ communicates_from_to(m, declare, last_comments (ag:agenda_item))

RI10 If the Chairperson has declared an agenda item closed, and not all items have been treated, then the Chairperson will announce one of the remaining items as the current item.
Formal
$\forall$t, i $\forall$m: CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed)

& ∃i agenda_item_at(γ, i, t) & not addressed_agenda_item_at(γ, i, t)
$\Rightarrow$ ∃t′ $\geq$ t state(γ, t′, output(m)) $\models$ communicates_from_to(m, declare, opened(i))

Exception Handling:
If the Chairperson closes the meeting,
Then the Chairperson has communicated to the participants that any remaining items on the agenda are postponed.

∀t, i ∀m: CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed)
$\Rightarrow$
∀i [agenda_item_at(γ, i, t) & not addressed_agenda_item_at(γ, i, t) $\Rightarrow$
∃ t′$\leq$t state(γ, t′, output(m)) $\models$ communicates_from_to(m, postpone, agenda_item(i)]

RI11 If the Chairperson has declared the meeting opened,
then the Chairperson will announce the proposed end time.
Formal ∀t, i ∀m:CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, meeting_opened)
$\Rightarrow$ ∃t′ $\geq$ t, t″ state(γ, t′, output(m)) $\models$ communicates_from_to(m, declare, planned_end_time(t″))

RI12 If the Chairperson has announced the proposed end time, then the Chairperson will open an agenda item.
Formal
∀t, t ∀m:CHAIR, ∃i
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, planned_end_time(t″))
$\Rightarrow$ ∃t′ (t $\leq$ t′ $<$ t″ state(γ, t′, output(m)) $\models$ communicates_from_to(m, declare, opened(i))

RI13 If the Chairperson has proposed an end time,
then the Chairperson will announce the agenda items.
∀t, t″, i ∀m:CHAIR
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, planned_end_time(t″)) & agenda_item_at(γ, i, t)
$\Rightarrow$ ∃t′ (t $\leq$ t′ $<$ t″ state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, agenda_item(i))

RI14 If the Chairperson has announced all agenda items,
then the Chairperson will ask if any Participant has another agenda item.
∀t, t″, ∀m:CHAIR
∀i [agenda_item_at(γ, i, t) $\Rightarrow$
∃t′ $<$ t state(γ, t′, output(m)) $\models$ communicates_from_to(m, declare, agenda_item(i))]
$\Rightarrow$ ∃t″$\geq$t state(γ, t″, output(m)) $\models$ communicates_from_to(m, request, other_items)

RI15 If the Chairperson has declared the last agenda item closed
then the Chairperson will close the meeting within 10 seconds.
∀t, t″, ∀m:CHAIR

$\forall$i [agenda_item_at(γ, i, t) $\Rightarrow$
$\exists$t$' \leq$ t state(γ, t$'$, output(m)) $\models$ communicates_from_to(m, declare, closed(i))] &
$\exists$i [agenda_item_at(γ, i, t) &
state(γ, t, output(m)) $\models$ communicates_from_to(m, declare, closed(i))] &
$\Rightarrow \exists$t$''$ (t $\leq$ t$'' <$ t + 10)
state(γ, t$''$, output(m)) $\models$ communicates_from_to(m, declare, meeting_closed)

Acknowledgments The authors are grateful to the editor and anonymous reviewers who gave a number of constructive suggestions on how to improve the paper and its readabilty, and to the participants in the meetings used as empirical material.

References

Barringer H, Fisher M, Gabbay D, Owens R, Reynolds M (1996) The imperative future: principles of executable temporal logic. Research Studies Press Ltd John Wiley & Sons

Bosse T, Jonker CM, van der Mey L, Treur J (2005) LEADSTO: a language and environment for analysis of dynamics by SimulaTiOn. In: Eymann T et al (eds) Proc. of the third german conference on multi-agent system technologies, MATES'05 vol 3550. Lecture Notes in Artificial Intelligence, Springer Verlag, pp 165–178

Carley KM, Gasser L (1999) Computational organization theory in multiagent systems: a modern approach to distributed artificial intelligence. Chapter 7. Gerhard Weiss (ed). MIT Press

Clarke EM, Grumberg O, Peled DA (2000) Model checking. MIT Press

Cornelissen F, Jonker CM, Treur J (2003) Compositional verification of knowledge-based task models and problem solving methods. Knowl Inf Syst J 5:337–367

Creighton JL, Using group process techniques to improve meeting effectiveness. URL: http://www. effectivemeetings.com/teams/teamwork/creighton.asp

Croston JD, Goulding HB (1966) The effectiveness of communication at meetings: a case study. Oper Res Quarterly 17(1):47–57

van Eck PAT, Engelfriet J, Fensel D, van Harmelen F, Venema Y, Willems M (2001) A survey of languages for specifying dynamics: A knowledge engineering perspective. IEEE Trans Knowl Data Eng 13:462–496

Fasli M (2003) Formal systems and agent-based social simulation equals null? J Artif Society Simul 7(4). Available at: <http://jasss.soc.surrey.ac.uk/7/4/7.html>

Fisher M (1994) A survey of concurrent MetateM—the language and its applications. In: Gabbay DM, Ohlbach HJ (eds) Temporal logic—Proceedings of the first international conference, Lecture notes in AI, vol 827, pp 480–505

Forbus KD (1984) Qualitative process theory. Artif Intell 24(1–3):85–168

Gamboa R, Kaufmann M (2001) Nonstandard analysis in ACL2. J Autom Reason 27:323–351

Garfinkel H (1967) Studies in ethnomethodology. Englewood Cliffs, NJ, Prentice-Hall

Goffman E (1961) Encounters. The Bobs-Merrill Company

Goffman E (1963) Behaviour in public places. Collier-MacMillan, London

Henzinger T, Nicollin X, Sifakis J, Yovine S (1994) Symbolic model checking for real-time systems. Inform Comput 111(2):193–244. Academic Press.

Jonker CM, Treur J (2002) Compositional verification of multi-agent systems: A formal analysis of proactiveness and reactiveness. In: de Roever WP, Langmaack H, Pnueli A (eds) Proceedings of the international workshop on compositionality, COMPOS'97. Lecture Notes in Computer Science, vol. 1536, Springer Verlag, 1998, pp. 350–380. Extended version in: International J Coop Infor Syst 11:51–92

Kelso JAS (1995) Dynamic patterns: the Self-organisation of brain and behaviour. MIT Press, Cambridge, Mass.

Law AD, Kelton WD (2000) Simulation modeling and analysis. McGraw Hill

Manna Z, Pnueli A (1995) Temporal verification of reactive systems: Safety. Springer Verlag

Moss S, Gaylard H, Wallis S, Edmonds B (1998) SDML: a multi-agent language for organizational modelling. Comput Math Organ Theory 4(1):43–70

Niederman F, Beise CM, Beranek PM (1996) Issues and concerns about computer-supported meetings: The facilitator's perspective. MIS Quarterly 20(1):1–22

Port RF, van Gelder T (eds) (1995) Mind as motion: explorations in the dynamics of cognition. MIT Press, Cambridge, Mass.

Reiter, R (2001) Knowledge in action: logical foundations for specifying and implementing dynamical systems. MIT Press, 2001
Robert HM (2000) Robert's rules of order (Newly Revised), 10th edn, HarperCollins Publishers
de Roever WP, Langmaack H, Pnueli A (eds) (1998) Proceedings of the international workshop on compositionality, COMPOS'97. Lecture Notes in Computer Science, vol. 1536, Springer Verlag
Serman CB, Basili VR (1998) Communication and organization: an empirical study of discussion in inspection meetings. IEEE Trans Soft Eng 24(6):559–572
Stirling C (2001) Modal and temporal properties of processes. Springer Verlag
Wolf K (2002) The makings of a good meeting. Available at: http://members.dcn.org/kjwolf

Catholijn M. Jonker is a full professor in Man Machine Interaction of the Delft University of technology in The Netherlands. She studied computer science at Utrecht University. She completed her PhD on the topic of Negations and Constraints in Logic Programming at the same university. After completing a post-doc position on the same topic at the Universität Bern, she became an assistant professor at the Vrije Universiteit Amsterdam and switched her research topic to agent technology. During the time at the Vrije Universiteit her interest in cognitive science increased, which she combined with her work on modeling multi-agent systems and organizations and her work on the analysis and modeling of the dynamics of behavior of complex systems. After working for the Vrije Universiteit she became a full professor in Artificial Intelligence and Cognitive Science at the Nijmegen Institute for Cognition and Information of the Radboud Universiteit Nijmegen in The Netherlands. First in Nijmegen and now in Delft she continues her research along the same lines.

Martijn C. Schut is Assistant Professor at the Department of Artificial Intelligence, Vrije Universiteit, Amsterdam, The Netherlands. He received a MSc from the Vrije Universiteit (NL) and a PhD from the University of Liverpool (UK). His research interests concern the emergence of organizational dynamics in distributed multi-agent systems.

Jan Treur received his Ph.D. in Mathematics and Logic in 1976 from Utrecht University. Since 1986 he works in Artificial Intelligence, from 1990 as a full professor and head of the Department of Artificial Intelligence at the Vrije Universiteit Amsterdam. In the 1990s he headed a research program on component-based design of knowledge-based and agent systems. In the last five years the research program focused on modeling dynamics of agent systems in practical application areas, and related to other disciplines such as Biology, Cognitive Science, Organization Theory, and Philosophy of Mind.

Pınar Yolum is an assistant professor at Boğaziçi University, Istanbul. She received her PhD and MS in computer science from North Carolina State University in 2003 and 2000, respectively, and her BSc in computer engineering from Marmara University, Istanbul in 1998. She worked as a post-doctoral researcher at the Vrije Universiteit Amsterdam. Her research interests include multiagent systems and service-oriented computing.